Limiting Partial Combinatory Algebras (Towards new interaction between category theory and proof theory)

AUTHOR(S):

Akama, Yohji

CITATION:

Akama, Yohji. Limiting Partial Combinatory Algebras (Towards new interaction between category theory and proof theory). 数理解析研究所講究録 2001, 1217: 1-22

ISSUE DATE:

2001-06

URL:

http://hdl.handle.net/2433/41228

数理解析研究所講究録 1217 巻 2001 年 1-22

Limiting Partial Combinatory Algebras *

Yohji Akama(赤間 陽二)
Mathematical Institute, Tohoku University †

March 31 2001

Abstract

We will construct from every partial combinatory algebra (PCA, for short) $\mathcal{A}$ a PCA a-lim($\mathcal{A}$) such that (1) every representable numeric function $\varphi(n)$ of a-lim($\mathcal{A}$) is exactly of the form $\lim_t \xi(t,n)$ with $\xi(t,n)$ being a representable numeric function of $\mathcal{A}$, and (2) $\mathcal{A}$ can be embedded into a-lim($\mathcal{A}$) which has a synchronous application operator. Here, a-lim($\mathcal{A}$) is $\mathcal{A}$ equipped with a limit structure in the sense that each element of a-lim($\mathcal{A}$) is the limit of a countable sequence of $\mathcal{A}$-elements. We will discuss limit structures for $\mathcal{A}$ in terms of Barendregt's range property. Moreover, we will repeat the construction lim(−) transfinite times to interpret infinitary λ-calculi. Finally, we will attempt to interpret type-free $\lambda\mu$-calculus by introducing another partial applicative structure which has an asynchronous application operator and allows a parallel limit operation.

1 Introduction

Partial combinatory algebras (PCA, for short) are partial applicative structures axiomatized by the same axioms as combinatory algebras, except that the application operators can be partial operators. The PCAs are important in connection with the realizability interpretations of intuitionistic logics. The realizability interpretations extract the computational content from intuitionistic logic's proofs as programs. Using PCAs to carry out the realizability interpretations, we can obtain 'realizability' models of typed term calculi and constructive set theories in which we can do mathematics. Cf. tripos construction(Hyland-Johnstone-Pitts[8]).

Recently, a new realizability interpretation was introduced by Nakata-Hayashi[11] to extract the computational content of semi-classical logic's proofs as approximation algorithms.

*The author acknowledges Susumu Hayashi, Mariko Yasugi, Stefano Berardi, and Ken-etsu Fujita.

† 東北大学 大学院理学研究科 数学専攻

They first noticed that Gold's limiting recursive functions[7], which was originally introduced to formulate the learning processes of machines, serve as approximation algorithms. Here, Gold's limiting recursive function is of the form $f(x)$ such that

$$f(x) = y \iff \exists t_0 \forall t > t_0.\, g(t,x) = y \iff \lim_t g(t,x) = y,$$

where $g(t,x)$ is called a guessing function, and t is a limit variable. Then, they proved that some limiting recursive functions approximate a realizer of a semi-classical principle $\neg\neg\exists y \forall x.\, g(x,y) = 0 \to \exists y \forall x.\, g(x,y) = 0$. Also, they showed impressive usages of the semi-classical principle for mathematics and for software synthesis.

In this way, Nakata-Hayashi opened up the possibility that limiting operations provide realizability interpretation of semi-classical logical systems.

They formulated the set of the limiting recursive functions as a *Basic Recursive Function Theory*(BRFT, for short. Wagner[19] and Strong[16]). Then Nakata-Hayashi carried out their realizability interpretation using the BRFT.

If we can formulate the set of limiting algorithms as a PCA $\mathcal{L}$, then by carrying out Nakata-Hayashi realizability interpretation using $\mathcal{L}$, we may be able to construct 'realizability' models of

1. semi-classical typed term calculi(e.g., typed Parigot's $\lambda\mu$-calculus, and typed term calculi with control operators),
2. semi-classical constructive set theories, and
3. constructive set/type theories with limit operations.

Motivated by the above, we will introduce a construction from a given PCA $\mathcal{A}$ another PCA a-lim($\mathcal{A}$). Our idea is

- limit variable t is the clock of the processing element(MPU).
- guessing function $g(t,x)$ is a generator of an infinite stream $\langle g(0,x), g(1,x), \ldots \rangle$.
- limiting recursive function $\lim_t g(t,x)$ is the stream modulo a symmetric, transitive relation $\sim$.

  Two streams are related with $\sim$ if for all natural number t except finite numbers, the two t-th elements of the two streams have the same value.

In a-lim($\mathcal{A}$), every allowed limit is exactly of the form $\lim_t a_t$ such that the infinite sequence $\langle\, a_t \in \mathcal{A} \,\rangle_t$

1. is indexed by $\mathbb{N}$; and
2. is generated inside the PCA $\mathcal{A}$.

We will call this $\lim_t a_t$ an autonomous limit. Owing to the above (1,2), we will be able to prove that every representable partial function of a-lim($\mathcal{A}$) is exactly

a limiting recursive function guessed by some representable partial functions of $\mathcal{A}$ (see Section 3).

Based on this result, we have only to take $\mathcal{L}$ as a-lim($\mathbb{N}$), in order to find realizability models of both semi-classical constructive set/type theories and constructive set/type theories which have limit structures.

In order to construct from the $\mathcal{A}$ a PCA with stronger computation power, we will first consider the following two forms of limits $\lim_\lambda a_\lambda$.

(R) $\langle\, a_\lambda \in \mathcal{A} \,\rangle_\lambda$ is indexed by the whole $\mathcal{A}$.

(N) $\langle\, a_\lambda \in \mathcal{A} \,\rangle_\lambda$ is any countable sequence of $\mathcal{A}$-elements.

We will prove that (R) does not always strengthen the computation power of $\mathcal{A}$. However, (N) has an extreme effect on the strength of the $\mathcal{A}$. We will introduce another construction from any PCA $\mathcal{A}$ to another PCA n-lim($\mathcal{A}$) where only allowed limit is of the form (N). Then, the set of representable functions in n-lim($\mathcal{A}$) is the set of all partial numeric functions. Moreover, it can compute a discontinuous function from $\mathbb{R}$ to $\mathbb{N}$. The construction n-lim($-$) may be interesting itself since it applies for all signature of partial algebras. See Section 4.

By using our results on limits over PCAs, we aim to interpret the following infinite λ-calculi. Infinite λ-calculi have been studied in proof- and recursion-theoretic contexts and are now being studied in the analysis of infinite streams(for input/output) and non-terminating recursive calls of functional programming languages.

1. Tait's typed calculus of infinitely long terms[17]: An infinite sequence of type A terms is again a type A term. His motivation was proof-theoretic. He wanted to define a large class of calculable functionals of finite types.

2. S. Feferman's typed calculus T_0 of infinitely long terms (For details, see Schwichtenberg and Wainer [13]): An infinite sequence $\langle P_1, P_2, \ldots \rangle$ of type A terms is again a type A term, if there is a term that calculates for each i the code of P_i. After Feferman developed Tait's typed λ-calculus in a proof-theoretic context, he introduced T_0 and studied T_0 in a recursion-theoretic context.

3. two infinite type-free λ-calculi of Kennaway-Klop-Sleep-de Vries[9] and Berarducci-Dezani[3]: Both have terms representing infinite Böhm-, Lévy-Longo-, or Berarducci-trees.

4. The type-free $\lambda\mu$-calculus (Parigot [12]): The typed version corresponds to the classical logic, and a typed/type-free version relates to a typed/type-free functional programming language with control operators such as `call/c`. Type-free $\lambda\mu$-calculus has μ-variables to represent continuations. By regarding μ-variables as infinite sequences of usual variables, we can regard $\lambda\mu$-calculus as an infinite λ-calculus.

The relationship between Tait's infinite typed λ-calculus and Feferman's typed λ-calculus is comparable to the relationship between n-lim$(-)$ and a-lim$(-)$

The infinite λ-calculi (1,2,3) have an infinite term consisting of infinite terms, while our constructions a-lim$(\mathcal{A})$ and n-lim$(\mathcal{A})$ introduce an element infinitely depending on elements which are "finite"(i.e., in $\mathcal{A}$). In order to interpret the infinite λ-calculi, we will repeat our constructions a-lim$(-)$ and/or n-lim$(-)$. The resulting PCA allows repeated limit $\lim_{t_1} \cdots \lim_{t_k} f(t_1, \ldots, t_k)$. See Section 5.

However, Parigot's type-free $\lambda\mu$-calculus is difficult to interpret with a-lim and/or n-lim. We will introduce another construction n-LIM$(-)$ which extends a given PCA $\mathcal{A}$ to a partial applicative structure n-LIM$(\mathcal{A})$ such that

- every allowed limit in n-LIM$(\mathcal{A})$ is a parallel limit $\lim_{t_1,\ldots,t_k} f(t_1, \ldots, t_k)$.
- the application operator is asynchronous.

This construction n-LIM$(-)$ was found by trial-and-error. With the help of concurrency theory, a branch of computer science, we will try to clarify the parallelism hidden in the parallel limits of n-LIM$(-)$. Then, we will interpret the type-free $\lambda\mu$-calculus in Section 6

Throughout this paper, the symbol "$\simeq$" means "if one-side is defined, then the other side is defined as having the same value," while the symbol "$=$" means that "both sides are defined as having the same value." The symbol "$\downarrow$" means "is defined." So, $t \downarrow$ is equivalent to an equation $t = t$. For any operation f, we assume that $f(a_1, \ldots, a_n) \downarrow$ implies $a_1 \downarrow, \ldots$ and $a_n \downarrow$. The set of partial functions from A to B is denoted by $A \rightharpoonup B$. We write $\overset{\infty}{\forall} x \in A.\, \varphi(x)$ to express that "$\{x \in A \mid \neg\varphi(x)\}$ is a finite set."

Organization of This Paper. In the next section, we consider limiting recursive functions which are guessed by partial recursive functions, while Gold's limiting recursive functions by total recursive functions. We calibrate the computational power of our limiting recursive functions in terms of arithmetical hierarchy of recursion theory.

In Section 3, we extend an arbitrary PCA with a limit structure. Specifically, we construct from every PCA $\mathcal{A}$ another PCA a-lim$(\mathcal{A})$ such that the representable partial functions of a-lim$(\mathcal{A})$ is exactly guessed by representable partial functions of $\mathcal{A}$.

In Section 4, we examine two possible limit structures for PCAs. By using Barendregt's range property, we derive that the first structure turns out to be useless. The second limit structure increases the computational power of PCAs. Specifically, we construct from every PCA $\mathcal{A}$ another PCA n-lim$(\mathcal{A})$ where every partial numeric function is representable. The construction n-lim$(-)$ applies every partial algebra.

In Section 5, we iterate transfinite times two constructions a-lim$(-)$ and n-lim$(-)$. We calibrate the set of representable partial numeric functions.

In Section 6, we construct an interpretation of type-free affine version of Parigot's $\lambda\mu$-calculus. The calculus has μ-variables which corresponds to continuations of a programming language Scheme. We regard μ-variables as streams of usual term variables.

In Section 7 and Section 8, we review related work and we conjecture some problems.

2 Limiting Recursion

Definition 2.1 We say a partial numeric function $\varphi(n_1, \ldots, n_k)$ is *guessed* by a partial numeric function $\xi(t, n_1, \ldots, n_k)$ as t goes to infinity, provided that $\forall n_1, \ldots, n_k \exists t_0 \forall t > t_0.\ \varphi(n_1, \ldots, n_k) \simeq \xi(t, n_1, \ldots, n_k)$. We write $\varphi(n_1, \ldots, n_k) \simeq \lim_t \xi(t, n_1, \ldots, n_k)$. For every class $\mathcal{F}$ of partial numeric functions, $\lim(\mathcal{F})$ denotes the set of partial numeric functions guessed by a partial numeric function in $\mathcal{F}$.

In calibrating the computational power of $\lim(\mathcal{F})$, we recall the *limiting recursive functions* introduced by Gold[7]. We assume the knowledge about the arithmetical hierarchy of sets and complete sets with respect to many-one reducibility. The standard reference is Soare's book[14].

Proposition 2.2 (Gold[7]) 1. A total function guessed by a partial recursive function can be guessed by a primitive recursive function.

2. A (partial) function guessed by a total recursive function is exactly a (partial) recursive function in the halting set $\mathcal{K}$ (called a *limiting recursive function* by Gold.)

We denote the set of partial recursive functions by **PRF**. We have $\lim(\mathbf{PRF}) \supseteq \mathbf{PRF}$, because every partial recursive function $\varphi(n_1, \ldots, n_k)$ is guessed by the partial recursive function $\varphi(\pi_2^{k+1}(t, n_1, \ldots, n_k), \ldots, \pi_{k+1}^{k+1}(t, n_1, \ldots, n_k))$, where each π_i^{k+1} returns the i-th argument. Below, we will show that $\lim(\mathbf{PRF})$ strictly includes the set of Gold's limiting recursive functions.

Proposition 2.3 (Takeshi Yamazaki, or folklore) 1. For every total function, it is guessed by a total recursive function if and only if the graph of the function is in a class Δ_2^0 of the arithmetical hierarchy.

2. For every partial function, if it is guessed by a partial recursive function, then the graph and the domain are both in a class Σ_3^0 of the arithmetical hierarchy. Moreover, there is a partial function f such that f is guessed by a partial recursive function and the graph of f and the domain of f are both complete with respect to many-one reducibility.

Proof. (1) $\lim_t g(t,x) = y$ iff $\exists t_0 \forall t > t_0 . g(t,x) = y$ iff $\forall t_0 \exists t > t_0 . g(t,x) = y$, because $\lim_t g(t,x)$ is always defined. (2) Let $\{n\}(-)$ be the unary partial recursive function which index is n, and $\mathcal{W}_n$ be the domain of $\{n\}(-)$. It is well-known that $\mathbf{Cof} = \{n \mid \mathcal{W}_n \text{ is cofinite}\} = \{n \mid \exists s \forall t > s . t \in \mathcal{W}_n\}$ is Σ^0_3 complete. Let the guessing function be $\xi(t,n) = 1$ if $t \in \mathcal{W}_n$, *undefined* otherwise. Then $x \in \mathbf{Cof} = \mathrm{dom}(\lim_t \xi(t,n))$ iff $(x,1) \in \mathrm{graph}(\lim_t \xi(t,n))$. Q.E.D.

The operation $\lim(-)$ has a good property, as Nakata-Hayashi showed. We will review one of their result.

We recall *ω-basic recursive function theory with a successor function* of Strong [16] and Wagner [19]. A class of partial numeric functions is called a *ω-basic recursive function theory with a successor function*(ω-BRFT with suc, for short), if it has for each n the enumeration functions $\varphi_n(e, x_1, \ldots, x_n)$, the S^m_n-functions, the successor, the constant functions, the projections, and the discriminator, and is closed under the composition.

Theorem 2.4 (Nakata-Hayashi[11]) If $\mathcal{F}$ is an ω-BRFT with suc, then so is $\lim(\mathcal{F})$.

Proof. The enumeration and S^m_n functions of $\lim(\mathcal{F})$ are naturally derived from those of $\mathcal{F}$. We should be careful in proving that $\lim(\mathcal{F})$ is closed under the composition. If the return value of $\varphi(x) \simeq \lim_t \xi(t,x)$ at x is undefined, then the return value of $\varphi'(\varphi(x))$ at x should be undefined for every $\varphi' \in \lim(\mathcal{F})$. Let $\varphi'(x) \simeq \lim_t \xi'(t,x)$. Unfortunately, $\lim \xi'(t, \xi(t,x))$ can be defined in case that ξ' is a constant function. But, $\varphi'(\varphi(x)) \simeq \lim_t \mathtt{if} \xi(t,x) = \xi(t+1,x)$ `then` $\xi'(t, \xi(t,x))$ `else` $\xi(t,x)$. Q.E.D.

3 Autonomous Limit

A *partial combinatory algebra(*PCA*, for short)* is $\mathcal{A} = \langle |\mathcal{A}|, \cdot, s, k \rangle$ such that $\cdot$ is a binary partial operator(*application operator*) on a set $|\mathcal{A}|$, and $s, k \in |A|$ are distinct elements subject to $(k \cdot a) \cdot b = a$, $(s \cdot a) \cdot b \downarrow$, $((s \cdot a) \cdot b) \cdot c \simeq (a \cdot c) \cdot (b \cdot c)$. Examples of PCA are the set $\mathbb{N}$ of natural numbers and the set of values of the call-by-value λ-calculus. $\mathcal{A}, \mathcal{B}, \ldots$ range over PCAs.

Given a PCA, we can simulate a λ-abstraction; for a "polynomial" $t[x]$, there is a "polynomial" $\lambda x . t[x]$ such that $(\lambda x . t[x]) \cdot a \simeq t[a]$. $\lambda x . a = k \cdot a$, $\lambda x . x = s \cdot k \cdot k$. $\lambda x . t[x] \cdot t'[x] \simeq s \cdot (\lambda x . t[x]) \cdot (\lambda x . t'[x])$. The *Church numeral* of a natural number n, denoted by $\overline{n}$, is a polynomial $\lambda y \lambda x . y \cdot (\cdots (y \cdot x) \cdots)$ with the y successively applied n-times to x.

We say a partial numerical function $\psi(t, n_1, \ldots, n_k)$ is *represented* by an element $a \in \mathcal{A}$, whenever $(\cdots((a \cdot \overline{t}) \cdot \overline{n_1}) \cdots) \cdot \overline{n_k} = \overline{m}$ if and only if $\psi(t, n_1, \ldots, n_k) = m$. The set of partial numerical functions representable in $\mathcal{A}$ is denoted by $\mathrm{RpFn}(\mathcal{A})$. It is well-known that $\mathrm{RpFn}(\mathcal{A})$ contains **PRF**.

Given a PCA $\mathcal{A}$, we will construct a PCA a-lim($\mathcal{A}$) such that $\mathrm{RpFn}(\text{a-lim}(\mathcal{A})) = \lim(\mathrm{RpFn}(\mathcal{A}))$.

Definition 3.1 (Autonomous Limit of PCA) Given a PCA $\mathcal{A} = \langle |\mathcal{A}|, \cdot, s, k \rangle$ *The extension* a-lim($\mathcal{A}$) *of* $\mathcal{A}$ *by the autonomous limits* is a PCA a-lim($\mathcal{A}$) = $\langle |\text{a-lim}(\mathcal{A})|, *, \mathbf{s}, \mathbf{k} \rangle$, where

- $|\text{a-lim}(\mathcal{A})|$ is a quotient set $\{a \in |\mathcal{A}| \mid a \sim a\}/\sim$, where $a \sim b$ is defined as $\overset{\infty}{\forall} t \in \mathbb{N}.\, (a \cdot \bar{t} = b \cdot \bar{t})$.
- $\mathbf{s} = [k \cdot s]_\sim, \mathbf{k} = [k \cdot k]_\sim$ ("s, k for any value t", t being "time");
- $[a]_\sim * [b]_\sim = [(s \cdot a) \cdot b]_\sim$ ("synchronous application").

As $\sim$ is a symmetric, transitive relation on $|\mathcal{A}|$, the quotient set $|\text{a-lim}(\mathcal{A})|$ is well-defined. The element is an equivalence class $[a]_\sim$ with $a \in \mathcal{A}$. When a is undefined, so will $[a]_\sim$ be undefined. The operation $*$ is well-defined; Suppose $a \sim a'$ and $b \sim b'$ and $s \cdot a \cdot b \cdot \overline{n} \simeq (a \cdot \overline{n}) \cdot (b \cdot \overline{n})$ is defined for values of n which are large enough. Then, because $a \cdot \overline{n} = a' \cdot \overline{n}$ and $b \cdot \overline{n} = b' \cdot \overline{n}$, we have $s \cdot a \cdot b \cdot \overline{n} \simeq (a \cdot \overline{n}) \cdot (b \cdot \overline{n}) \simeq (a' \cdot \overline{n}) \cdot (b' \cdot \overline{n}) \simeq s \cdot a' \cdot b' \cdot \overline{n}$. Therefore, $[a] * [b] \simeq [a'] * [b']$.

Theorem 3.2 a-lim($\mathcal{A}$) is a PCA such that RpFn(a-lim($\mathcal{A}$)) = lim(RpFn($\mathcal{A}$)).

Proof. Let $t \in \mathbb{N}$ be sufficiently large.

- $\mathbf{s} * [a] * [b] * [c] \simeq [s(s(s(ks)a)b)c]$ while $[a] * [c] * ([b] * [c]) \simeq [s(sac)(sbc)]$. An axiom for s implies $s(s(s(ks)a)b)c\bar{t} \simeq (ks\bar{t})(a\bar{t})(b\bar{t})(c\bar{t})$. By using the axiom for k with $\bar{t}\downarrow$, the last is $\simeq s(a\bar{t})(b\bar{t})(c\bar{t})$. It is, by an axiom for s, $\simeq s(sac)(sbc)\bar{t}$. So, $\mathbf{s} * [a] * [b] * [c] \simeq [a] * [c] * ([b] * [c])$.
- $\mathbf{s} * [a] * [b] \simeq [s(s(ks)a)b]$. An axiom of s implies $s(s(ks)a)b\bar{t} \simeq ks\bar{t}(a\bar{t})(b\bar{t})$. By using the axiom of k with $\bar{t}\downarrow$, it is $\simeq s(a\bar{t})(b\bar{t})$. It is always defined because $a\bar{t}\downarrow$, $b\bar{t}\downarrow$ and an axiom of s. So, $\mathbf{s} * [a] * [b]\downarrow$.
- $\mathbf{k} * [a] * [b] = [s(s(kk)a)b]$ and $s(s(kk)a)b\bar{t} \simeq (kk\bar{t})(a\bar{t})(b\bar{t})$. By using the axiom of k with $\bar{t}\downarrow$, it is $\simeq k(a\bar{t})(b\bar{t})$. It is $\simeq a\bar{t}$ as $b\bar{t}\downarrow$. So, $\mathbf{k} * [a] * [b] = [a]$.

Therefore, a-lim($\mathcal{A}$) is indeed a PCA.

Before proving the latter part of the theorem, we note that a Church numeral $\overline{\mathbf{n}}$ of a-lim($\mathcal{A}$) is $[k\overline{n}]_\sim$ with the latter Church numeral $\overline{n}$ being in $\mathcal{A}$.

Both of RpFn(a-lim($\mathcal{A}$)) and lim(RpFn($\mathcal{A}$)) have the nowhere defined partial functions of any arity. Let φ be a unary partial function somewhere defined. Then, $\varphi \in \text{RpFn(a-lim}(\mathcal{A}))$ iff $\exists a \in \mathcal{A}(a \sim a \;\&\; \forall n, m\ (\ [a]_\sim * \overline{\mathbf{n}} = \overline{\mathbf{m}}$ iff $\varphi(n) = m\ ))$ iff $\exists a \in \mathcal{A}(\overset{\infty}{\forall} t.\, a\bar{t}\downarrow \;\&\; \forall n, m(\overset{\infty}{\forall} t \in \mathbb{N}.\, sa(k\overline{n})\bar{t} = (k\overline{m})\bar{t})$ iff $\varphi(n) = m)$ iff

$$\exists a \in \mathcal{A}(\overset{\infty}{\forall} t.\, a\bar{t}\downarrow \;\&\; \forall n, m(\overset{\infty}{\forall} t \in \mathbb{N}.\, a\bar{t}\overline{n} = \overline{m})\ \text{iff}\ \varphi(n) = m\ ) \qquad (1)$$

On the other hand, $\varphi \in \text{lim(RpFn}(\mathcal{A}))$ is equivalent to

$$\exists \xi \in \text{RpFn}(\mathcal{A}) \forall n, m(\lim_t \xi(t, n) = m \text{ iff } \varphi(n) = m) \qquad (2)$$

As φ is somewhere defined, we have $\varphi(l)\downarrow$ for some l. Therefore, $\overset{\infty}{\forall}\, t.\, \xi(t,l)\downarrow$. So, every $a \in \mathcal{A}$ representing ξ satisfies $\overset{\infty}{\forall}\, t.\, a\bar{t}\downarrow$. Thus, (1) if and only if (2). Therefore, $\varphi \in \mathrm{RpFn}(\text{a-lim}(\mathcal{A}))$ if and only if $\varphi \in \lim(\mathrm{RpFn}(\mathcal{A}))$. For φ with the arity being greater than 1, it is similarly proved. Q.E.D.

Definition 3.3 (homomorphism) A function f from a PCA $\mathcal{A}$ to a PCA $\mathcal{B}$ is a *homomorphism*, if f preserves the operators as relations. That is, $f(s) = \mathbf{s}$, $f(k) = \mathbf{k}$, and $ab = c$ implies $f(a)f(b) = f(c)$.

We denote by $\mathbb{PCA}$ the category of PCAs and homomorphisms between them. An injective, surjective homomorphism is called an *isomorphism*. We abbreviate a homomorphism by homo.

Theorem 3.4 Define the function $\iota_{\mathcal{A}}$ by $\iota_{\mathcal{A}} : \mathcal{A} \to \text{a-lim}(\mathcal{A});\ a \mapsto [k \cdot a]_{\sim}$. Then,

1. $\iota_{\mathcal{A}}$ is an injective, non-surjective homo.
2. $[a]_{\sim} = \iota_{\mathcal{A}}(b) \iff \overset{\infty}{\forall}\, t \in \mathbb{N}.\ a \cdot \bar{t} = b \quad (\iff \text{“}\lim_{t}\ a \cdot \bar{t} = b\text{”})$
3. $a \cdot a' = b \iff \iota_{\mathcal{A}}(a) * \iota_{\mathcal{A}}(a') = \iota_{\mathcal{A}}(b)$.

Proof. (1) $\iota_{\mathcal{A}}$ is clearly an injective homo. If $[skk]_{\sim} = \iota_{\mathcal{A}}(b)$, then $\overset{\infty}{\forall}\, t.\, skk\bar{t} \simeq kb\bar{t}$, which is $\overset{\infty}{\forall}\, t.\, \bar{t} \simeq b$. Contradiction. So, $\iota_{\mathcal{A}}$ is not surjective. (2) and (3) are trivial. Q.E.D.

4 Possible Limit Structures

Each element of the PCA a-lim($\mathcal{A}$) is of the form $\lim_t a_t$, where the

1. parameter t can be any natural number; and
2. the sequence $\langle a_t \rangle_t$ is of the form $\langle a \cdot \bar{t} \rangle_t$ for some $a \in \mathcal{A}$. In this case, the sequence is “autonomously tracked” by an $\mathcal{A}$-element a.

To justify the necessity of the two conditions, we will discuss following alternative $\lim_t a_t$ for $\mathcal{A}$: **(R)** t of $\lim_t a_t$ can be any element of $\mathcal{A}$ (See Subsection 4.1); or **(N)** the sequence $\langle a_t \rangle_t$ is any countable sequence.(See Subsection 4.2)

4.1 Range property and limit.

We consider the following semantics of lim operation:

$$\lim_{a} f(\vec{x}, a) = y \iff \overset{\infty}{\forall}\, a \in |\mathcal{A}|.\ y = f(\vec{x}, a). \tag{3}$$

Although this limit is useful in a PCA $\mathbb{N}$, it is not the case for the set $\mathcal{M}$ of closed λ-terms modulo β-equality. $\mathcal{M}$ is called a closed term model of $\lambda\beta$-calculus.

Even though we can construct from $\mathcal{M}$ another PCA $A(\mathcal{M})$ with the limit being (3), we have $\mathbf{PRF} = \mathrm{RpFn}(A(\mathcal{M})) = \mathrm{RpFn}(\mathcal{M}) \neq \lim(\mathrm{RpFn}(\mathcal{M})) = \lim(\mathbf{PRF})$. This is because of the range property studied by Barendregt([1, p.517]): In $\mathcal{M}$, the range of any combinator is either a singleton or an infinite set.

Indeed, for any definable function $f : \mathcal{M} \times \mathcal{M} \to \mathcal{M}$, so is the function $F_x(a) = f(x, a)$. When $\lim_a f(x, a)$ in the above sense has a value, F_x has the finite range $\{F_x(a) \mid a \in \mathcal{M}\}$. However, it must be a singleton because of the range property. Therefore, the lim of (3) will be useless.

4.2 Limit along all the countable sequences.

Theorem 4.1 Let $\mathcal{M}$ be a partial algebra of a signature $\langle f_1, \ldots;\ u_1 = u_1', \ldots;\ v_1 \simeq v_1', \ldots\rangle$. Here $f_1, \ldots$ are (partial) operators, and $u_1 = u_1', \ldots; v_1 \simeq v_1', \ldots$ are axioms. Each u_i, u_i', v_i, v_i' is built up from the variables and the operators $f_1, \ldots$.

The extension n-lim($\mathcal{M}$) *of* $\mathcal{M}$ *by the non-constructive limits* is $\langle |\text{n-lim}(\mathcal{M})|,\ \mathbf{f}, \ldots\rangle$ such that

1. $|\text{n-lim}(\mathcal{M})| = \{a : \mathbb{N} \rightharpoonup \mathcal{M} \mid a \sim a\}/\sim$, where $a \sim b$ is defined as $\overset{\infty}{\forall}\, k \in \mathbb{N}.\, a(k) = b(k)$.
2. For $\mathbf{c}^{(k)} = [\ c^{(k)}\ ]_\sim \in |\text{n-lim}(\mathcal{M})|$, define $\mathbf{f}(\mathbf{c}^{(1)},\ \mathbf{c}^{(1)}, \ldots) \simeq [\ \varphi\ ]_\sim$ with $\varphi(i) \simeq f(\ c^{(1)}(i),\ c^{(2)}(i), \ldots)$.

Then n-lim($\mathcal{M}$) is a partial algebra of the same signature.

As we defined homomorphisms for PCAs, we will define a homomorphism for partial algebras as a function which preserves the operators as relations.

Theorem 4.2 Define the function $\iota_{\mathcal{M}} : \mathcal{M} \to \text{n-lim}(\mathcal{M});\ b \mapsto [a]_\sim$, where $a(t) = b$. Then,

1. $\iota_{\mathcal{M}}$ is an injective homo. If $\mathcal{M}$ has at least two elements, it is non-surjective.
2. $[a]_\sim = \iota_{\mathcal{M}}(b) \iff (\overset{\infty}{\forall}\, t \in \mathbb{N}.\ a(t) = b) \qquad (\iff \text{“}\lim_t a(t) = b\text{”})$
3. $\mathbf{f}(b, \ldots) = b'$ in $\mathcal{M} \iff \iota_{\mathcal{M}}(\mathbf{f})(\iota_{\mathcal{M}}(b), \ldots) = \iota_{\mathcal{M}}(b')$ in n-lim($\mathcal{M}$).

Proof. Assume $\iota_{\mathcal{M}}$ is surjective. Then, let $a : \mathbb{N} \rightharpoonup \mathcal{M}$ such that $a(t) = b_{t \bmod 2}$ with distinct $b_0, b_1 \in \mathcal{M}$. We have $\exists b\ \overset{\infty}{\forall}\, t \in \mathbb{N}\,(a(t) = b)$. Contradiction. The other claims are clear. Q.E.D.

In computer science, partial algebras appear as algebraic specification of software. For instance, stacks. It is indeed partial, because `pop(nil)` can be undefined. The signature of stacks is $\langle$`pop, push, nil, 1; pop(push(x, y)) = x`$\rangle$. By taking $\mathcal{M}$ of above theorems as PCAs, we will have the following:

Theorem 4.3 If $\mathcal{A}$ is a PCA, n-lim($\mathcal{A}$) is a PCA such that RpFn(n-lim($\mathcal{A}$)) is the set of all partial numeric functions.

Proof. We will prove only the second part. Since any partial numeric function of finite domain is representable in any PCA, every partial numeric m-ary function φ is represented by $[f]_\sim$ such that $f(t)$ represents $\varphi \restriction \{0,1,\ldots,t-1,t\}^m$ in $\mathcal{A}$. Q.E.D.

Remark 4.4 There are other PCAs where every partial numeric function is representable. For instance, D_∞ introduced by Scott and the partial continuous operations (PCO, for short. See [18, Ch.9, Sect.4]) introduced by Kleene. However, no n-lim($\mathcal{A}$) is isomorphic to them. This fact is explained by comparing the three extensional collapses[18] of n-lim($\mathcal{A}$), D_∞ and PCO.

On the one hand, the extensional collapses of both D_∞ and PCO are the same; a type structure consisting of all total Kleene-Kreisel continuous functionals over $\mathbb{N}$. By a Kleene-Kreisel continuous functional, we mean that the output of the functional depends only on finite subinformation of the input functionals.

On the other hand, the extensional collapse of n-lim($\mathcal{A}$) contains a discontinuous functional over $\mathbb{N}$. One such example is the Gauss function, a function given a real number x returning the smallest integer not greater than x. The Gauss function is actually a functional over $\mathbb{N}$; every real number may be regarded as a converging sequence of rational numbers, and thus as a function from $\mathbb{N}$ to $\mathbb{Q}$. Because $\mathbb{Q}$ is encoded to $\mathbb{N}$, every real number is a function from $\mathbb{N}$ to $\mathbb{N}$. The Gauss function is not Kleene-Kreisel continuous, because the output needs infinite precision on the input real number to determine whether the input is an integer or not. As Gauss function is rewritten as the limit according to Yasugi-Brattka-Washihara[20], it is in the extensional collapse of n-lim($\mathcal{A}$).

So, n-lim($\mathcal{A}$) is isomorphic to neither D_∞ nor PCO.

5 Repeating Limits

In functional programming, we use infinite lists as streams(for input/output) and non-terminating recursive calls. These objects are usually the unwinding("limit") of finite objects. To analyze such infinite objects, of interest are *infinite* λ-calculi which were introduced by Kennaway-Klop-Sleep-de Vries[9] and Berarducci-Dezani[3]. Both calculi admit a term like $\lambda x.y^\omega x$, $\lambda x.\, y^\omega(x^\omega)$, and have terms with the limit operation $(-)^\omega$ being nested.

Interpretations of these infinite λ-calculi may be constructed by our constructions a-lim$(-)$ and n-lim$(-)$ of PCAs. In such cases, it is necessary to repeat the limit constructions ω-times.

Definition 5.1 (α-times repeated limits) For every PCA $\mathcal{A}$ and every ordinal number α, let us define a PCA $\lim^\alpha(\mathcal{A})$ and the canonical injective homo. $\iota^\alpha_\beta : \lim^\beta(\mathcal{A}) \to \lim^\alpha(\mathcal{A})$ for each $0 < \beta \leq \alpha$.

- a-lim$^0(\mathcal{A}) = \mathcal{A}$, and ι^β_β is the identity for each ordinal number β.
- a-lim$^{\beta+1}(\mathcal{A})$ = a-lim $\Big($a-lim$^\beta(\mathcal{A})\Big)$, and $\iota^{\beta+1}_\gamma = \iota_{\text{a-lim}^\beta(\mathcal{A})} \circ \iota^\beta_\gamma$.

- When α is a limit ordinal number,
  a-lim$^{\alpha}(\mathcal{A})$ is an inductive limit of $\langle \iota^{\delta}_{\gamma} \rangle_{0<\gamma<\delta<\alpha}$, and each ι^{α}_{δ} is a natural injection of the inductive limit.

We similarly define n-lim$^{\alpha}(\mathcal{A})$.

Theorem 5.2 For each ordinal number $\alpha > 0$,

1. a-lim$^{\alpha}(\mathcal{A})$ is indeed a PCA, and ι^{α}_{β} is an injective, non-surjective homo. for $\beta < \alpha$;
2. RpFn(a-lim$^{\alpha}(\mathcal{A})$) is lim$^{\alpha}$(RpFn($\mathcal{A}$)) for α finite, and is $\bigcup_{\beta \in \mathbf{N}}$ lim$^{\beta}$ (RpFn($\mathcal{A}$)) otherwise.

Proof. The first part will proved by transfinite induction on α. In proving the second part of the theorem, note that RpFn(a-lim$^{\alpha}(\mathcal{A})$) is $\bigcup_{\beta<\alpha}$ RpFn(a-lim$^{\beta}(\mathcal{A})$). In view of Theorem 3.2, it is already proved for finite α. For $\alpha = \omega + 1$, RpFn(a-lim$^{\omega+1}(\mathcal{A})$) is lim $\left(\bigcup_{\beta<\mathbf{N}} \text{RpFn(a-lim}^{\beta}(\mathcal{A}))\right)$, which is lim $\left(\bigcup_{\beta<\mathbf{N}} \text{lim}^{\beta}(\text{RpFn}(\mathcal{A}))\right)$. Therefore, $\varphi(-) \in$ RpFn(lim$^{\omega+1}(\mathcal{A})$) iff $\exists \beta < \omega \exists \xi \in \text{lim}^{\beta}(\text{RpFn}(\mathcal{A}))\forall n.\ \exists t'_0 \forall t_0 > t'_0\ \varphi(n) \simeq \xi(t_0, n)$. Hence, $\varphi(-) \in$ RpFn(lim$^{\omega+1}(\mathcal{A})$) iff $\exists \beta < \omega \exists \xi' \in \text{RpFn}(\mathcal{A})\ \exists t'_0 \forall t_0 > t'_0 \cdots \exists t'_{\beta} \forall t_{\beta} > t'_{\beta}\ \varphi(n) \simeq \xi'(t_0, \cdots, t_{\beta}, n)$ Thus, RpFn(lim$^{\omega+1}(\mathcal{A})$) is RpFn(lim$^{\omega}(\mathcal{A})$). In this way, we can prove the second part of the theorem. Q.E.D.

Theorem similar to Theorem 3.4 also holds

We do not know whether there is an ordinal number α such that a-lim$^{\alpha}(\mathcal{A})$ is isomorphic to a-lim (a-lim$^{\alpha}(\mathcal{A})$). $\iota^{\alpha+1}_{\alpha}$ cannot be such an isomorphism, according to above Theorem. Although the construction a-lim$(-)$ is an endofunctor of a category $\mathbb{PCA}$, it is difficult to employ a category-theoretic version of Tarski's fixpoint theorem arguments; because two homo's a-lim$(\iota^{\beta+1}_{\beta})$ and $\iota^{\beta+2}_{\beta+1}$ are not equal but their equalizer can be proved to be $\iota^{\beta+1}_{\beta}$.

Lemma 5.3
1. a-lim$(-)$ will be an endofunctor on $\mathbb{PCA}$ by defining a-lim$(f)([a]_{\sim}) = [f(a)]_{\sim}$. The functor a-lim preserves injective homomorphisms.
2. $\iota^{\beta+1}_{\beta}$ is an equalizer of $\iota^{\beta+2}_{\beta+1}$ and a-lim$(\iota^{\beta+1}_{\beta})$.
   In other words,

$$\forall x \in \text{a-lim}^{\beta+1}(\mathcal{A})\ (\iota^{\beta+2}_{\beta+1}(x) = \text{lim}(\iota^{\beta+1}_{\beta})(x) \iff \exists y \in \text{a-lim}^{\beta}(\mathcal{A})(x = \iota^{\beta+1}_{\beta}(y))).$$

Proof.

1. a-lim(f) is well-defined as a function, that is, if $a \sim b$ then $f(a) \sim f(b)$. Indeed, As f is a homomorphism, we have $f(a) \cdot \overline{n} \simeq f(a) \cdot f(\overline{n}) \simeq f(a \cdot \overline{n})$, and similarly $f(b) \cdot \overline{n} \simeq f(b \cdot \overline{n})$. If $a \sim b$, then $\overset{\infty}{\forall}\ n \in \mathbb{N}$, we have that $a \cdot \overline{n} \simeq b \cdot \overline{n}$ and that $f(a) \cdot \overline{n} \simeq f(a \cdot \overline{n}) \simeq f(b \cdot \overline{n}) \simeq f(b) \cdot \overline{n}$. As for the homomorphic property of a-lim(f), we only show that f preserves an intrinsic constant k: we have a-lim$(f)(k) \overset{Def.3.1}{\simeq}$ a-lim$(f)([k \cdot k]) \overset{the\ def}{\simeq}$

$[f(k \cdot k)] \overset{f \text{ homo}}{\simeq} [k \cdot k] \overset{Def.3.1}{\simeq} k$. It is easy to see that a-lim preserves the composition operation and the identity.

2. $x = [a]_\sim$ for some a. By (1), we have $\text{a-lim}(\iota_\beta^{\beta+1})(x) = [\iota_\beta^{\beta+1}(a)]_\sim = [[k \cdot a]_\sim]_\sim$ and $\iota_{\beta+1}^{\beta+2}(x) = [k \cdot [a]_\sim]_\sim = [[k \cdot k]_\sim \cdot [a]_\sim]_\sim = [[s \cdot (k \cdot k) \cdot a]_\sim]_\sim$. Therefore
$$\iota_{\beta+1}^{\beta+2}(x) = \text{a-lim}(\iota_\beta^{\beta+1})(x) \tag{4}$$
is equivalent to $[k \cdot a]_\sim \sim [s \cdot (k \cdot k) \cdot a]_\sim$ in $\text{a-lim}^{\beta+1}(\mathcal{A})$. This is equivalent to that
$$\overset{\infty}{\forall} n \in \mathbb{N}\big([k \cdot a]_\sim \cdot \overline{n} \simeq [s \cdot (k \cdot k) \cdot a]_\sim \cdot \overline{n}.\big) \tag{5}$$
We have the left-hand side of (5) $\simeq [s \cdot (k \cdot a) \cdot (k \cdot \overline{n})]_\sim$, and the right-hand side $\simeq [s \cdot (s \cdot (k \cdot k) \cdot a) \cdot (k \cdot \overline{n})]_\sim$. Therefore (5) is equivalent to that
$$\overset{\infty}{\forall} n \in \mathbb{N}, \overset{\infty}{\forall} m \in \mathbb{N}\ \big(s \cdot (k \cdot a) \cdot (k \cdot \overline{n}) \cdot \overline{m} \simeq s \cdot (s \cdot (k \cdot k) \cdot a) \cdot (k \cdot \overline{n}) \cdot \overline{m}\big) \tag{6}$$
Because the left-hand side of (6) $\simeq a \cdot \overline{n}$ and the right-hand side $\simeq (s \cdot (k \cdot k) \cdot a \cdot \overline{m}) \cdot \overline{n} \simeq k \cdot (a \cdot \overline{m}) \cdot \overline{n} \simeq a \cdot \overline{m}$, (6) is equivalent to that
$$\overset{\infty}{\forall} n \in \mathbb{N}\ \overset{\infty}{\forall} m \in \mathbb{N} a \cdot \overline{n} \simeq a \cdot \overline{m} \tag{7}$$
This means that a sequence $\langle a \cdot \overline{n} \rangle_n$ converges to some y. So, (6) is equivalent to that $\exists y\ \overset{\infty}{\forall} n \in \mathbb{N}. a \cdot \overline{n} \simeq y$. That is, $a \sim k \cdot y$. Therefore, $x = [a]_\sim = [k \cdot y]_\sim = \iota(y)$. Therefore, (4) is equivalent to $\exists y.\ x = \iota_\beta^{\beta+1}(y)$.

(2) Let f be an injective homomorphism and let $\text{a-lim}(f)([x]_\sim) = \text{a-lim}(f)([y]_\sim)$. By the definition, $f(x) \sim f(y)$ Thus $\overset{\infty}{\forall} n \in \mathbb{N}. f(x) \cdot \overline{n} \simeq f(y) \cdot \overline{n}$. As f is a homomorphism, $\overset{\infty}{\forall} n \in \mathbb{N}. f(x \cdot \overline{n}) \simeq f(y \cdot \overline{n})$. As f is injective, $\overset{\infty}{\forall} n \in \mathbb{N}. x \cdot \overline{n} \simeq y \cdot \overline{n}$. Therefore $x \sim y$, which is $[x]_\sim = [y]_\sim$. Q.E.D.

Theorem 5.4 For each ordinal number $\alpha > 0$,

1. $\text{n-lim}^\alpha(\mathcal{A})$ is indeed a PCA, and ι_β^α is an injective, non-surjective homo. for $\beta < \alpha$;
2. $\text{RpFn}(\text{n-lim}^\alpha(\mathcal{A}))$ is the set of partial numeric functions.

An infinitely long term $\langle P_1, P_2, \ldots \rangle$ of Tait's and Feferman's λ-calculi respectively can be interpreted with $\text{n-lim}^{\alpha+1}(\mathcal{A})$ and $\text{a-lim}^{\alpha+1}(\mathcal{A})$ respectively as follows:
$$\langle a_1, a_2, a_3, \ldots, \rangle \simeq [t \mapsto \langle a_1, a_2, a_3, \ldots, a_t \rangle_t]_\sim,$$
where a_i is an interpretation of the term P_i in $\text{a-lim}^\alpha(\mathcal{A})$ and $\text{n-lim}^\alpha(\mathcal{A})$ respectively, and the ordinal number α depends on the complexity of P_i's. Here, $\langle a_1, a_2, a_3, \ldots, a_t \rangle$ is the abbreviation of $\mathbf{cons}\ a_1\ (\cdots(\mathbf{cons}\ a_t\ \mathbf{i})\cdots)$, for a pairing $\mathbf{car}\ (\mathbf{cons}\ a\ b) = a$, and $\mathbf{cdr}\ (\mathbf{cons}\ a\ b) = b$. For example, we can define
$$\mathbf{cons} = \lambda xyz.\, xyz, \quad \mathbf{car} = \lambda x.\, x\mathbf{k}, \quad \mathbf{cdr} = \lambda x.\, x(\mathbf{ki}). \tag{8}$$

Remark 5.5 *All the morphisms which appear so far are happensto be discrete, projective, decidable applicative morphisms of Longley [10]. According to [10, Chapter 3], they all induce functors between realizability toposes and they all preserve discrete objects; projectives; and finite colimits, NNO.*

6 An Interpretation of Type-free $\lambda\mu$-calculus

In [12], Parigot introduced the typed $\lambda\mu$-calculus which corresponds to classical propositional logic via Curry-Howard isomorphism. By forgetting the types in the $\lambda\mu$-calculus, we obtain a *type-free $\lambda\mu$-calculus*. Both calculi are related to type-free/typed programming languages with control operators (call/cc, $\mathcal{C}$ introduced by M. Felleisen and D. Friedman in [4], raise-handle, ...).

Since Nakata-Hayashi interpret a weak classical logic by a limiting realizability interpretation, it is natural to ask whether we can interpret a $\lambda\mu$-calculus by a λ-model which has a limit structure. We will concentrate on the type-free version of $\lambda\mu$-calculus.

Type-free $\lambda\mu$-calculus is specified by defining $\lambda\mu$-terms, and the reduction rules (the β-reduction rule and the μ-reduction rule).

$\lambda\mu$-Terms are generated by $M ::= c \mid x \mid MM \mid \lambda x.\, M \mid [\alpha]M \mid \mu\alpha.\, M$. An occurrence of α in $\mu\alpha.\cdots$ will be called a *bound* occurrence of α. An occurrence of α which is not bound is called *free*. A $\lambda\mu$-term $[\alpha]M$ is regarded as application of M to the continuation bound to α. $\mu\alpha.\, M$ is regarded as functional abstraction over the continuation variable α on the level of continuation semantics. Here, $x, y, z, \ldots$ range over term-variables. $\alpha, \beta, \gamma, \ldots$ range over *μ-variables* which are distinguished from the term-variables.

Mixed substitution. A *context* is generated by the above grammar with a special constant ( ). By replacing the occurrences of ( ) of a context $C(\,)$ with a $\lambda\mu$-term M, we obtain a $\lambda\mu$-term $C(M)$.

For any context $C(\,)$ and a $\lambda\mu$-term N, we define a *mixed substitution* $\vartheta = [[\alpha](\,) := C(\,)]$ as follows; If N is a $\lambda\mu$-term, so is a $N\vartheta$. If N does not have a free μ-variable α, then $N\vartheta$ is N. So, if N is a variable or $\mu\alpha.\, M$, then $N\vartheta \equiv N$. The mixed substitution commutes with a λ-abstraction and an application. The mixed substitution $\vartheta = [[\alpha](\,) := C(\,)]$ satisfies $([\beta]M)\vartheta \equiv [\beta](M\vartheta)$ and $(\mu\beta.\, M)\vartheta \equiv \mu\beta.\, (M\vartheta)$, provided $\beta \neq \alpha$. Finally, $([\alpha]M)[[\alpha](\,) := C(\,)] \equiv C(M[\,[\alpha](\,) := C(\,)\,])$.

μ-reduction is specified by the following rule: $(\mu\alpha.\, M)N \to_\mu \mu\alpha.\Big(M[[\alpha](\,) := [\alpha]((\,)N)]\Big)$. In graphical notation, the rule is

$$(\mu\alpha.(\cdot\,([\alpha]P)\,\cdot))N \to_\mu \mu\alpha.(\cdot\,([\alpha](P'N))\,\cdot). \tag{9}$$

For instance, $(\mu\alpha.\,[\alpha]\,(y[\alpha]x))z \to_\mu \mu\alpha.\,[\alpha]\,(y\,([\alpha](xz))\,z)$. Of cou
free $\lambda\mu$-calculus has the usual β-reduction.

6.1 Informal Semantics of $\lambda\mu$

**μ-application(abstraction,reduction) = infinite applicatio
β-reduction).** Consider an informal translation from the type-fr
to (infinitely long) type-free λ-calculus:

$$[\alpha]P \mapsto \tilde{P}\alpha_0 \ldots \alpha_m \ldots \qquad \text{and} \qquad \mu\alpha.\,M \mapsto \lambda\alpha_0 \ldots \alpha_m \ldots$$

Then, the above rewriting rule (9) is translated to

$$(\lambda\alpha_0\alpha_1 \ldots (\cdot\cdot(\tilde{P}\alpha_0\alpha_1 \ldots)\cdot\cdot))\tilde{N} \to_\beta \lambda\alpha_0\alpha_1 \ldots (\cdot\cdot(\tilde{P}'\tilde{N}\alpha_0\alpha_1 \ldots$$

which turns out to be a β-reduction between infinite terms, if w
bound variables infinite times $\lambda\alpha_1 \ldots.\ (\cdot\cdot\cdot(\tilde{P}'\tilde{N}\alpha_1 \ldots)\cdot\cdot) \equiv$
$(\tilde{P}'\tilde{N}\alpha_0\alpha_1 \ldots)\cdot\cdot)$. The η-like reduction on continuation $\mu\alpha.\,[\alpha]P \to$
turns out to be just *infinite* η-reductions(see Theorem 6.14).

This idea of Parigot is being studied by Fujita[5]; with typ
translation above precisely corresponds to Gödel translation and
$\alpha_0, \alpha_1, \ldots$ is finite.

μ-variable = infinite stream. The idea of (10) leads us to in

$$[\alpha]P \simeq \lim_t \tilde{P}\alpha_0\alpha_1 \ldots \alpha_t \simeq [t \mapsto \tilde{P}\alpha_0\alpha_1 \ldots \alpha_t]_\sim \quad \alpha_i \simeq \mathbf{car}{*}\overbrace{(\mathbf{cdr}\,*$$

Then, we have a *Swap rule* of Streicher-Reus's version of type-free
$[\mathbf{cons}{*}\ M\ {*}N]P \simeq [N](P{*}M)$, if we allow more general *continuati*
mere μ-(continuation) variables namely pure λ-terms stacked t
$\mathbf{cons} * M_1 * (\mathbf{cons} * M_2 * \ldots * (\mathbf{cons} * M_n * \alpha)\ldots)$.

Before we consider the interpretation of μ-abstraction, we v
rigorous.

μ-reduction causes delay in a stream. The translation resul
ing rule (9) is $f \to g$ such that

$$\begin{aligned} f(t) &\simeq (\lambda a_0 \ldots a_t.\,(\cdot\cdot(\tilde{P}a_0 \ldots a_t)\cdot\cdot))\tilde{N} \simeq \lambda a_1 \ldots a_t.\,(\cdot\cdot(\tilde{P}\tilde{N}a_2 \\ g(t) &\simeq \lambda a_0 \ldots a_t.(\cdot\cdot(\tilde{P}\tilde{N}a_0 \ldots a_t)\cdot\cdot) \end{aligned}$$

But, $f(t+1) \simeq g(t)$. Because it takes 1 'clock time' to compute(b
g from f, a delay will occur because of the extra computational
Anyway, $[f]_\sim \neq [g]_\sim$ in n-lim($\mathcal{A}$). So, n-lim($\mathcal{A}$) does not interpre
Neither does a-lim($\mathcal{A}$).

To equate f and g above, let us replace the symmetric, transi
with the smallest symmetric, transitive relation $\approx$ containing $\sim$ a

rule $f \approx (t \mapsto f(t+1))$. Unfortunately, a quotient set $(\mathbb{N} \rightharpoonup |\mathcal{A}|) \ / \approx$ cannot have the synchronous application operator $[f]_\approx * [g]_\approx \simeq [t \mapsto f(t) \cdot g(t)]_\approx$ well-defined.

6.2 Asynchronous Applicative Structure and Parallel Limit

Given a PCA $\mathcal{A}$, we introduce another partial algebra n-LIM($\mathcal{A}$) where an appropriate application operator can be defined.

The carrier set of n-LIM($\mathcal{A}$) is $\{f \mid \exists n \geq 0.\, f : \mathbb{N}^n \rightharpoonup |\mathcal{A}| \text{ and } f \sim f\}/\sim$, where $\sim$ is a symmetric, transitive relation over $\bigcup_{n\geq 0}(\mathbb{N}^n \rightharpoonup |\mathcal{A}|)$ defined by the symmetricity rule, the transitivity rule plus the following two rules.

1. The 'delay' rule. $\mathbb{N}^n \stackrel{f}{\rightharpoonup} \mathcal{A} \sim \mathbb{N}^n \xrightarrow{id\times\cdots\times id\times suc\times id\cdots\times id} \mathbb{N}^n \stackrel{f}{\rightharpoonup} |\mathcal{A}|$ where id is the identity function on $\mathbb{N}$ and suc is the successor function.

   As for $\times$, it is an associative operator and for all $f_i : A_i \rightharpoonup B_i$ we have $f_1 \times f_2 : A_1 \times A_2 \rightharpoonup B_1 \times B_2$ such that $(f_1 \times f_2)(a_1, a_2)$ is $(f_1(a_1), f_2(a_2))$ if each $f_i(a_i)$ is defined, and it is undefined otherwise.

   The rule is necessary to have $\lim_t f(t) \simeq \lim_t f(t+1)$.

2. The 'exchange' and 'weakening' rule.

$$\mathbb{N}^n \stackrel{f}{\rightharpoonup} |\mathcal{A}| \sim \mathbb{N}^m \xrightarrow{(\pi^m_{\sigma(1)}, \pi^m_{\sigma(2)}, \cdots, \pi^m_{\sigma(n)})} \mathbb{N}^n \stackrel{f}{\rightharpoonup} |\mathcal{A}|$$

   where $m \geq n$, σ is a permutation on $\{1, \ldots, n\}$, and for each $k = 1, \ldots, m$ the function π^m_k returns the k-th argument.

   For all $f_i : A \rightharpoonup B_i$ we have $(f_1, \ldots, f_n) : A \rightharpoonup B_1 \times \cdots \times B_n$ such that $(f_1, \ldots, f_n)(a)$ is $(f_1(a), \ldots, f_n(a))$ if each $f_i(a)$ is defined, and it is undefined otherwise.

   The rule will make the following application operator well-defined.

Lemma 6.1 Let $i, j, k \geq 0$. If $\overset{\infty}{\forall}\, (u_1, \ldots, u_k) \in \mathbb{N}^k.\ \forall s_1, \ldots, s_i, t_1, \ldots, t_j \in \mathbb{N}.$ $f_1(u_1, \ldots, u_k, s_1, \ldots, s_i) \simeq f_2(u_1, \ldots, u_k, t_1, \ldots, t_j)$, then $f_1 \sim f_2$,

Proof. Assume that if $\vec{t} > t_0$, then $f_1(\vec{t}, \vec{t_1}) \simeq f_2(\vec{t}, \vec{t_2})$. Let $g_i(\vec{t}, \vec{t_i}) \simeq f_i(t_1 + t_0, \ldots, t_k + t_0, \vec{t_i})$. Then, $f_i \sim g_i$ by kt_0 times repeated applications of the 'delay' rule. We can derive $g_1 \sim g_2$ by the 'exchange' and 'weakening' rule from that $g_1(\vec{t}, \vec{t_1}) \simeq g_2(\vec{t}, \vec{t_2})$. By the transitivity of $\sim$, we have $f_1 \sim f_2$. Q.E.D.

Remark 6.2 Consistent with Lemma 6.1, for each X = a-lim($\mathcal{A}$), n-lim($\mathcal{A}$), or n-LIM($\mathcal{A}$), we mention the relationship between the symmetric, transitive relation $\sim$ for X and the limit structure of X.

- For a-lim($\mathcal{A}$), we have always $i = j = 0$ and $k = 1$ and for an autonomous sequence of $\mathcal{A}$-elements the sequential limit $\lim_t a \cdot \bar{t}$ corresponds to $[a]_\sim \in$ a-lim($\mathcal{A}$).

- For n-lim($\mathcal{A}$), we have always $i = j = 0$ and $k = 1$ and for a $f : \mathbb{N} \rightharpoonup |\mathcal{A}|$ the sequential limit $\lim_t f(t)$ corresponds to $[f]_\sim \in$ n-lim($\mathcal{A}$).
- For n-LIM($\mathcal{A}$), for a $f : \mathbb{N}^{k+i} \rightharpoonup |\mathcal{A}|$ the parallel limit of $\lim_{t_1,\ldots,t_{k+i}} f(t_1, \ldots, t_{k+i})$ corresponds to $[f]_\sim \in$ n-LIM($\mathcal{A}$).

As is common, the contraction rule is seen as a communication(synchronization). In defining $\sim$, we cannot replace the 'exchange' and 'weakening' rule with the 'exchange,' 'weakening' and 'contraction' rule:

$$\mathbb{N}^n \xrightarrow{f} |\mathcal{A}| \sim \mathbb{N}^m \xrightarrow{(\pi^m_{\sigma(1)}, \pi^m_{\sigma(2)}, \ldots, \pi^m_{\sigma(n)})} \mathbb{N}^n \xrightarrow{f} |\mathcal{A}|$$

where $m \geq n$, σ is a function on $\{1, \ldots, n\}$.

An asynchronous application operator. For $f : \mathbb{N}^n \rightharpoonup |\mathcal{A}|$ and $g : \mathbb{N}^m \rightharpoonup |\mathcal{A}|$, define

$$[f]_\sim * [g]_\sim \simeq [h]_\sim \text{ , with } h = \mathbb{N}^n \times \mathbb{N}^m \xrightarrow{f \times g} |\mathcal{A}| \times |\mathcal{A}| \xrightarrow{(-)\cdot(-)} |\mathcal{A}|$$

where $(-) \cdot (-)$ is the application operator of a given PCA $\mathcal{A}$. The operator $*$ is 'asynchronous' in the sense that $f \times g$ is involved. The operator, as well as the symmetric, transitive relation $\sim$, permits 'delay' in the arguments (as streams). Therefore,

Lemma 6.3 $(-) * (-)$ is well-defined.

We say n-LIM($\mathcal{A}$) is *the extension of a* PCA $\mathcal{A}$ *by the non-constructive parallel limits and the asynchronous application.*

Remark 6.4 We can explain the application operator with the vocabulary of the concurrency theory:

1. f and g (respectively) is a process having at most n and m (respectively) independent clocks. For all time slices[2] except for finite numbers, we can observe $\mathcal{A}$-elements.
2. h is a process having the clocks of both f and g. Given a time slice, let a be the observation of f at a given time slice and let b be the observation of g at a given time slice. Then the observation of h at a given time slice is $a \cdot b$.

Lemma 6.5 In n-LIM($\mathcal{A}$), we have $[k]_\sim * x * y = x$. It is not the case that $[s]_\sim * x * y * z \simeq x * z * (y * z)$. But, it is the case if $x = [h]_\sim$ with some $h \in \mathcal{A}$.

[1] A total differentiable function $f(x, y)$ of analysis and an analytic function $F(z)$ of complex variable $z = x + iy$ allow us to calculate the derivative f', F' by either an iterated sequential limit $\lim_x \lim_y \ldots$ or a parallel limit $\lim_{x,y} \ldots, \lim_z \ldots$. The variable x and y is cooperating and 'communicating' each other in the calculation of the derivative. If x, y can be seen as clocks, f, F as a process, and f', F' as the generator of the process, then the total differentiability and being analytic can be seen as a cooperating and/or communicating process.

[2] $(t_1, \ldots, t_n)$ where each t_i is the value of the clock.

Proof. Let $x = [f]_\sim, y = [g]_\sim, z = [h]_\sim$, and the arities of f, g, h be n, m, l. $[k]_\sim * x * y$ is $[p]_\sim$ such that $p = f \circ (\pi_1^{n+m} \times \cdots \times \pi_n^{n+m}) \sim f$ by the exchange and weakening rule. Therefore $[k]_\sim * x * y = x$.

$[s]_\sim * x * y * z$ is $[u]_\sim$ with u being naturally an $(n+m+l)$-ary partial function, while $x * z * (y * z)$ is $[v]_\sim$ with v being naturally $(n+l+m+l)$-ary. So, it is difficult to have the equation unless $l = 0$ (i.e., $h \in \mathcal{A}$). Q.E.D.

Definition 6.6 For every polynomial $t[x]$, define the polynomial $\lambda x.\, t[x]$ as follows. Let x not occur in u.

- $\lambda x.\, a \simeq [k]_\sim * a. \qquad \lambda x.\, x \simeq [\lambda x.\, x]_\sim, \qquad \lambda x.\, u * x \simeq u.$
- $\lambda x.\, t[x] * u \simeq [\lambda xyz.\, xzy]_\sim * (\lambda x.\, t[x]) * u.$
- $\lambda x.\, u * t[x] \simeq [\lambda xyz.\, x(yz)]_\sim * u * (\lambda x.\, t[x]).$
- $\lambda x.\, t[x] * t'[x] \simeq [s]_\sim * (\lambda x.\, t[x]) * (\lambda x.\, t'[x]).$

Lemma 6.7 For every $a \in$ n-LIM($\mathcal{A}$) we have $(\lambda x.\, t[x])a \simeq t[a]$, if x occurs in $t[x]$ at most once, or if $a = [h]_\sim$ with some $h \in \mathcal{A}$.

The *Church numeral* $\overline{t}$ below is defined with the λ-abstraction of Definition 6.6.

Lemma 6.8 n-LIM($\mathcal{A}$) has a pairing **cons, car, cdr**; and **nth** such that **nth** $*$ $\langle x_0, \ldots, x_n \rangle * \overline{t} \simeq x_t$.

Proof. The pairing is by Lemma 6.7 and (8). Let **nth** $\simeq \lambda xy.$ **car**$*(y*$**cdr**$*x)$. Q.E.D.

We can define similarly *the extension* a-LIM($\mathcal{A}$) *of a* PCA $\mathcal{A}$ *by the autonomous parallel limits and the asynchronous application.*

6.3 The Interpretation

Convention 1 In every $\lambda\mu$-term M, every bound μ-variable is distinct[3] and different from any free μ-variables.

Definition 6.9 Given a type-free $\lambda\mu$-term M. Let $\{\alpha, \beta, \ldots\}$ be the set of μ-variables in M, then a set $\{t_\alpha, t_\beta, \ldots\}$ of numeric variables is denoted by M^p.

We will define the partial function M^g returns at most one $\mathcal{A}$-element, when the values of M^p is determined; $x^g \simeq x$ ranges over the elements of $\mathcal{A}$, $(MN)^g \simeq M^g N^g$ and $(\lambda x.\, M)^g \simeq \lambda x.\, M^g$.

$$\begin{aligned} ([\alpha]M)^g &\simeq M^g(\mathbf{nth}\ \alpha\ \overline{0}) \ldots (\mathbf{nth}\ \alpha\ \overline{t_\alpha}); \\ (\mu\alpha.\, M)^g &\simeq \lambda\alpha_0 \ldots \alpha_{t_\alpha}.\, M^g[\alpha := \langle \alpha_0, \ldots, \alpha_{t_\alpha} \rangle]. \end{aligned}$$

The interpretation $[\![M]\!]$ of M in n-LIM($\mathcal{A}$) is $[M^g]_\sim$.

[3]That is, $\alpha \neq \beta$, if M is $\cdots(\mu\alpha.\, \cdots\alpha\cdots)\cdots(\mu\beta.\, \cdots\beta\cdots)\cdots$.

We will prove that this interpretation works well for the al
$\lambda\mu$-calculus.

Lemma 6.10 $(P[x := Q])^g \simeq P^g[x := Q^g]$.

Proof. By induction on P. We abbreviate $[x := Q]$ as θ, $[x :=$ (

1. Case P is $[\alpha]M$. Then the left hand side $([\alpha](M\theta))^g$ is $\simeq (M\theta$
   By induction hypothesis, it is $M^g\theta^g$ $(\mathbf{nth}\,\alpha\,\overline{0})$ ... $(\mathbf{nth}\,\alpha\,\overline{t_\alpha})$
   which is the right hand side.

2. Case P is $\mu\beta.\,M$. Then the left hand side is $(\mu\beta.\,(M\theta))^g \simeq \lambda\beta$
   $\langle\beta_0,\ldots,\beta_{t_\beta}\rangle]$. By induction hypothesis, it is $\lambda\beta_0\ldots\beta_{t_\beta}.\,M^g\theta$
   Because we can assume that β is not in Q without loss of g
   last is $(\lambda\beta_0\ldots\beta_{t_\beta}.\,M^g\ [\beta := \langle\beta_0,\ldots,\beta_{t_\beta}\rangle])\ \theta^g$, which is tl
   side.

When P is of the other forms, then it is trivial. Q.E.D.

Lemma 6.11 $(\ P[[\alpha](\ ) := (\ )x_0\ldots x_{t_\alpha}]\ )^g \simeq P^g[\,\alpha := \langle x_0,\ldots,x$

Proof. By induction on P. ϑ stands for $[\ [\alpha](\ ) := (\ )x_0\ldots x_{t_\alpha}\ ]$ and $[\,\alpha := \langle x_0,\ldots,x_{t_\alpha}\rangle\ ]$.

1. Case P is $[\alpha]M$. Then the left hand side is $(M\vartheta\ x_0\ldots x_{t_\alpha})^g$.
   hypothesis, it is $M^g\theta x_0\ldots x_{t_\alpha}$, which is the right hand side

2. Case P is $\mu\beta.\,M$ with $\beta \neq \alpha$. Then the left hand side
   which is $\lambda\beta_0\ldots\beta_{t_\beta}.\,(M\vartheta)^g\ [\beta := \langle\beta_0,\ \ldots\ \beta_{t_\beta}\rangle]$. By induc
   sis, it is $\lambda\beta_0\ldots\beta_{t_\beta}.\,M^g\theta\ [\beta := \langle\beta_0,\ \ldots\ \beta_{t_\beta}\rangle]$. It is $\lambda\beta_0\ldots$
   $\langle\beta_0,\ \ldots\ \beta_{t_\beta}\rangle]\theta$, which is the right hand side.

When P is of the other forms, then it is trivial. Q.E.D.

Theorem 6.12 If x occurs free at most once in M, or if no μ-v
in N, then $((\lambda x.\,M)N)^g \simeq (M[x := N])^g$. Hence, $[\![(\lambda x.\,M)N]\!] \simeq [$

Proof. The left hand side is $(\lambda x.\,M^g)N^g$, where x occurs free a
M^g, or $N \simeq [h]_\sim$ for some $h \in \mathcal{A}$. So, the left hand side is M^g
Lemma 6.7. It is the right hand side by Lemma 6.10. Q.E.D.

Theorem 6.13 Let α occur free at most once in P, or let no μ-
in Q. Then we have

μ-reduction.	$[(\mu\alpha.\,P)Q]$	$\simeq$	$[\mu\alpha.\,P[\,[\alpha](\,) := [\alpha](($
Parigot's S3 rule.	$[\mu\alpha.\,M]$	$\simeq$	$[\lambda x.\,\mu\alpha.\,M[\,[\alpha](-)\ :=$

Proof. In order to prove the two statements, we will claim respectively

1. if $f(t_\alpha, t_{\vec{\beta}}) \simeq ((\mu\alpha.\, P)Q)^g$, then $f(t_\alpha{+}1, t_{\vec{\beta}}) \simeq \Big(\mu\alpha.\, P[[\alpha](\,) := [\alpha]((\,)Q)]\Big)^g$.; and
2. if $f(t_\alpha, t_{\vec{\beta}}) \simeq (\mu\alpha.\, P)^g$, then $f(t_\alpha{+}1, t_{\vec{\beta}}) \simeq \Big(\lambda x.\, \mu\alpha.\, P[[\alpha](\,) := [\alpha]((\,)x)\,]\Big)^g$.

(1) The left hand side $f(t_\alpha{+}1, t_{\vec{\beta}})$ is $(\lambda\alpha_0 \ldots \alpha_{t_\alpha+1}.\, P^g[\alpha := \langle\alpha_0, \ldots, \alpha_{t_\alpha+1}\rangle\,])Q^g$, which is, by Lemma 6.11, $(\lambda\alpha_0 \ldots \,(\, P[\,[\alpha](\,) := (\,)\alpha_0 \ldots \alpha_{t_\alpha+1}\,]\,)^g)Q^g$, which is, by Lemma 6.10 and the premise, $\lambda\alpha_1 \ldots \alpha_{t_\alpha+1}.\, (\, P[\,[\alpha](\,) := (\,)Q\alpha_1 \ldots \alpha_{t_\alpha+1}\,]\,)^g$, which is, by renaming bound variables, $\lambda\alpha_0 \ldots \alpha_{t_\alpha}.\, (\, P[\,[\alpha](\,) := (\,)Q\alpha_0 \ldots \alpha_{t_\alpha}]\,)^g$. Because of Convention 1, the mixed substitution above is a composition of two mixed substitutions: $[\,[\alpha](\,) := [\alpha]((\,)Q)\,][\,[\alpha](\,) := (\,)\alpha_0 \ldots \alpha_{t_\alpha}\,]$. By Lemma 6.11, $f(t_\alpha + 1, t_{\vec{\beta}}) \simeq \lambda\alpha_0 \ldots \alpha_{t_\alpha}.\Big(P[[\alpha](\,) := [\alpha]((\,)Q)]\Big)^g\ [\alpha := \langle\alpha_0, \ldots, \alpha_{t_\alpha}\rangle]$, which is the right hand side $\Big(\mu\alpha.\, P[[\alpha](\,) := [\alpha]((\,)Q)]\Big)^g$. The claim (2) can be proved in a similar way. Q.E.D.

Theorem 6.14 (η_{cont}) If α is not free in M, then $(\,\mu\alpha.\, [\alpha]M\,)^g \simeq M^g$, hence $[\,\mu\alpha.\, [\alpha]M\,] \simeq [M]$.

Proof. The translation $(-)^g$ unwinds each occurrence α^i of α to the same sequence of usual variables $\alpha_0, \alpha_1, \ldots, \alpha_{t_\alpha}$. Q.E.D.

When we validate the μ-reduction but not η_{cont}-reduction, we can replace $(-)^g$ with another translation $(-)^G$ satisfying the following two conditions [4] .

1. For the same μ-variable α, we will distinguish the different occurrences by $\alpha^1, \alpha^2, \ldots$. let $(-)^G$ unwind α^i to $\alpha_0, \alpha_1, \ldots, \alpha_{t_{\alpha^i}}$
2. For the μ-reduction with the following graphical notation

   $$(\mu\alpha^1.(\cdots([\alpha^2]P)\cdots([\alpha^3]Q)\cdots))N \to_\mu \mu\alpha^1.(\cdots([\alpha^2](P'N))\cdots([\alpha^3]Q')\cdots),$$

   we have $t_1 \geq t_2, t_3, \ldots$.

7 Related Work

Model of Lambda-Mu Calculus. In [6], Fujita constructed a model of type-free $\lambda\mu$-calculus, which is induced, in a typed case, from Gödel-Gentzen translation. The term model is obtained by the use of a fixed point of a typable λ-term

[4] These two conditions may be expressed along the line of Remark 6.4 with the vocabulary of the concurrency theory (like, "$\mu\alpha$ waits the α's inside the scope.....").

Because we comply Convention 1, we assign to a given $\lambda\mu$-term the largest number of clocks. Renaming of bound μ-variables of a $\lambda\mu$-term M can save the number of clocks which is assigned to M. Renaming of bound μ-variables may correspond to a certain scheduling among the parallel execution. But, I don't know.

which corresponds to one side of a reduction rule associated with double negation elimination in a typed case. In fact, he present Gödel-Gentzen translation as

- $x^g = x$.
- $(M_1 M_2)^g = M_1^g M_2^g$.
- $(\lambda x.\, M)^g = \lambda x.\, M^g$.
- $(\mu\alpha.\, M)^g = \mathbf{C}(\lambda\alpha.\, M^g)$.
  Here, $\mathbf{C}$ is the Turing's fixpoint operator applied to $\lambda xyz.\, x(\lambda k.\, y(\lambda f.\, k(fz)))$.
- $([\alpha]M)^g = \alpha M^g$.

Then, $(\mu\alpha.\, M)^g$ is potentially infinite λ-abstraction; we have $(\mu\alpha.\, M)^g =_\beta \lambda x_1 \ldots x_n.\, (\mu\alpha.\, M)^g x_1 \ldots x_n$ for any n. So, his idea is similar to our idea (10). Because we are concerned with relating $\lambda\mu$ to some process calculus, we used stream(=limit) instead of fixpoint operator.

Classical Logic as Limit. We aim to bridge the gap between constructive and classical logic through (Nakata-Hayashi) realizability interpretations. In order to do so, we are concerned only with properties of limit operations which are relevant to PCAs.

Different use of limit in interpreting classical logic are introduced by Berardi. In [2], Berardi defined a constructive model for Δ_2^0 maps. His model refines constructive interpretations for classical reasoning over one-quantifier formulas. In his model, he used a completion idea, quite similar to the topological completion producing $\mathbb{R}$ out of $\mathbb{Q}$. He was concerned with the process to compute the limit value. Based on that processes, he directly interpreted his semi-formal system of Δ_2^0 maps.

The main difference is the following

1. He uses intuition reasoning, and, consequently, he uses cofinally true conditions in place of definite true condition (classically, they are the same for converging limits). Suppose that $l : D \to \mathbb{N}$ is a converging limit, and define $P(l)$ iff $P(l(d))$ cofinally on D. In his interpretation, he may turn every proof of $P(l)$ into a proof that $P(l(d))$ cofinally in d, and in particular, $P(l(d))$ for some $d \in D$. But the only way we have of finding it is to go through all possible values for d.

2. When we write $\lim_{t\to\infty} \xi(t, x)$, we think of t as the clock of some guessing function $\xi(t, x)$, which eventually (from some t_0) stabilizes to some limit value. He has our idea as particular case, but he rather prefers to think of t as the finite set $t = \{y_1, \ldots, y_n\}$ of y's he used to check the correctness of a guessing function $\xi(y, x)$. He supposes to be given a total ordering in the range of values of $\xi(t, x)$, as it is the case when ξ guesses the minimum witness of some excluded middle $\exists y.\, p(t, x) \vee \forall y.\, \neg p(t, x)$. This

realizers may be $\langle 2,0\rangle$ if $p(y,x)$ is false for all y, or $\langle 1,y\rangle$ if $p(y,x)$. He orders realizers lexicographically, and he let $\xi(t,x)$ be minimum realizer in $\{\xi(y,x) \mid y \in t\}$. As t increases (as a set), $\xi(t,x)$ eventually stabilizes, but differently from ours. He wants to represent computations in which he never needs to check his guess against all natural numbers.

8 Conjectures

1. We conjecture that there is a constructive set theory such that if we construct non-constructive limits n-lim$(-)$ inside it, then outside the theory the resulting limits are actually are autonomous limits a-lim$(-)$.

2. The realizability topos over a PCA $\mathcal{A}$ will be denoted by $\mathsf{RT}(\mathcal{A})$. We conjecture that for every morphism $f : X \times \mathbb{N} \to Y$ in $\mathsf{RT}(\mathcal{A})$, we have $\lim_t f(-,t) : X \to Y$ as a morphism of $\mathsf{RT}(\text{a-lim}(\mathcal{A}))$. Conversely, we conjecture that every morphism of $\mathsf{RT}(\text{a-lim}(\mathcal{A}))$ is obtained in this way. Remark 5.5 may be relevant.

Acknowledgment

The author acknowledges Susumu Hayashi and Mariko Yasugi for stimulating discussion on limiting computation for discontinuous functions on $\mathbb{R}$ and limiting realizability interpretation. He acknowledges Stefano Berardi for limiting interpretation for classical logic. The author acknowledges Ken-etsu Fujita for stimulating discussion on $\lambda\mu$-calculi and for continual encouragement.

References

[1] H.P. Barendregt. *The Lambda Calculus, Its Syntax and Semantics.* North-Holland, second edition, 1984.

[2] Stefano Berardi. Classical logic as limit completion I&II, 2001.

[3] A. Berarducci and M. Dezani-Ciancaglini. Infinite lambda-calculus and types. *Theoretical Computer Science*, 212:29–75, 1999.

[4] M. Felleisen and D.P. Friedman. Control operators, the SECD machine, and the λ-calculus. In M. Wirsing, editor, *Formal descriptions of programming concepts III*, pages 193–217. North-Holland, 1986.

[5] K. Fujita. On proof terms and embeddings of classical substructural logics. *Studia Logica*, 61(2):199–221, 1998.

[6] Ken-Etsu Fujita. Simple models of pur $\lambda\mu$-calculus. Extended Abstract, March 2001.

[7] E. Mark Gold. Limiting recursion. *Journal of Symbolic Logic*, 30:28–48, 1965.

[8] J. M. E. Hyland, P. T. Johnstone, and A. M. Pitts. Tripos theory. *Math. Proc. Cambridge Philos. Soc.*, 88(2):205–231, 1980.

[9] J.R. Kennaway, J.W. Klop, M.R. Sleep, and F.J. de Vries. Infinitary lambda calculus. *Theoretical Computer Science*, 175(1):93–125, 1997.

[10] J.R. Longley. *Realizability Toposes and Language Semantics.* PhD thesis, University of Edinburgh, 1994.

[11] M. Nakata and S. Hayashi. Limiting first order realizability interpretation. *Sci. Math. Japonicae*, 2000. submitted.

[12] Michel Parigot. $\lambda\mu$-calculus: an algorithmic interpretation of classical natural deduction. In *Logic programming and automated reasoning (St. Petersburg, 1992)*, pages 190–201. Springer, Berlin, 1992.

[13] H. Schwichtenberg and S. S. Wainer. Infinite terms and recursion in higher types. In J. Diller and G. H. Müller, editors, $\models$ *ISILC Proof Theory Symposion (Proc. Internat. Summer Inst. and Logic Colloq., Kiel, 1974)*, pages 341–364. Lecture Notes in Math., Vol. 500, Berlin, 1975. Springer. Dedicated to Kurt Schütte on the occasion of his 65th birthday, Lecture Notes in Mathematics, Vol. 500.

[14] R. I. Soare. *Recursively Enumerable Sets and Degrees.* Perspectives in Mathematical Logic. Springer-Verlag, Berlin, 1987.

[15] Th. Streicher and B. Reus. Classical logic, continuation semantics and abstract machines. *J. Funct. Programming*, 8(6):543–572, 1998.

[16] H.R. Strong. Algebraically generalized recursive function theory. *IBM Journal of Research and Development*, 12:465–475, 1968.

[17] W.W. Tait. Infinitely long terms of transfinite type. In Crossley and Dummet, editors, *Formal Systems and Recursive Functions*, pages 176–185. North-Holland, 1965.

[18] Anne Sjerp Troelstra and Dirk van Dalen. *Constructivism in Mathematics*, volume 123 of *Studies in Logic and the foundations of Mathematics.* North-Holland, 1988.

[19] E. G. Wagner. Uniformly reflexive structures: On the nature of Gödelizations and relative computability. *Transactions of the American Mathematical Society*, 144:1–41, 1969.

[20] M. Yasugi, V. Brattka, and M. Washihara. Computability properties of Gaußian function. available via http://www.kyoto-su.ac.jp/~yasugi/Recent/gaussjuly19.ps, 1998. Revized version of an article in Proc. of CCA'98,Brno.